NFSU JOURNAL OF CYBER SECURITY & DIGITAL FORENSICS

NFSU – Journal of Cyber Security and Digital Forensics
Volume – 4, Issue – 2, December 2025
E – ISSN – 2583-7559

Polymorphic Malware in Forensics: A Systematic Review of Tools and Detection Techniques

Ujjval Rana
National Forensic Sciences University, Gandhinagar
Ujjval.bsmsfs1212624@nfsu.ac.in

Dr. Krishna Modi
Assistant Professor, National Forensic Sciences University, Gandhinagar
krishna.modi@nfsu.ac.in

Dr. Kapil Shukla
Assistant Professor, National Forensic Sciences University, Gandhinagar
kapil.shukla@nfsu.ac.in

***Abstract*—** Polymorphic malware is one of the biggest challenges facing forensic cybercrime investigations because it can change its code structure while maintaining malicious functionality. The traditional signature-based tools will not be able to detect these ever-evolving threats. This scoping review explores some of the forensic-relevant techniques, tools, datasets, and machine learning approaches in the detection of polymorphic malware. A total of 42 studies met the inclusion requirements. This paper synthesizes static, dynamic, hybrid, and machine learning-based detection methods. It enumerates various detection tools like Ghidra, Radare2, Cuckoo Sandbox, ANY.RUN, and machine learning frameworks that include TensorFlow, PyTorch, and Scikit-learn. Benchmarks of performance, usage of datasets, forensic applicability, limitations, and directions for future research are summarized. While hybrid and ML-driven modern detection methods show promising accuracy, dataset bias, limited generalization, computational overhead, and forensic admissibility concerns remain challenges. Future work on standardization, cross-dataset validation, explainability, and forensic-integrated workflows is much needed.

Keywords*— *Malware Detection; Polymorphic Malware; Static Analysis; Dynamic Analysis; Machine Learning

I. INTRODUCTION

The function of evolving over time to avoid detection by signature based detection systems, polymorphic malware remains a top concern for modern cybercrime investigations. Unlike static malicious software, which does not alter its internal design once written, polymorphic malware changes its internal design, but maintains its original intent to be destructive. Examples of the different ways polymorphic malware can mutate include encryption engines, packers, instruction substitution, register re-assignment, code transposition and automated obfuscation techniques [11][10]. Because each new infection includes a completely new configuration of the same malware, all of these variations make it difficult for forensic investigators to accurately identify and analyze them [7][16]. In addition to compromising a network by spreading itself through multiple running mutated versions of itself at numerous endpoints, polymorphic malware can also compromise log files and hide from monitoring applications[14][15]. Compromising a network's logging capability causes a loss of correlation among the forensic evidence left behind by the malware, causes confusion in the forensic investigator's attempt to construct an accurate timeline of an attack, and creates ambiguity in the chain of custody of the evidence collected during the investigation. Therefore, because they cannot consistently locate similar traceable elements within the many evolving strain versions of the malware, forensic investigators are increasingly relying upon behavioral, semantic and machine learning types of analytical techniques to find commonalities between the various versions of the malware [22] [3].

In addition, the rapid growth of MaaS (Malware-as-a-service) type ecosystems and the availability of polymorphic engines, including UPX modification engine, PESpin engine, Themida engine and custom packer engines have increased the popularity of using polymorphism in cybercrime attacks. Modern cyberattackers now commonly employ polymorphic ransomware, polymorphic trojans, polymorphic cryptominers and polymorphic keyloggers to carry out large-scale attacks that circumvent traditional endpoint security measures. These types of attacks are increasing the burden on organizations' information technology departments and forensic teams to perform attribution of the attacks, document the incidents and report the findings legally.Therefore, to address this problem, several researches have studied various methods for performing analyses of polymorphic malware.

Traditionally, malware detection has been carried out using classic static analysis techniques: the review of PE headers, entropy distribution, opcode sequences, n-grams, import tables, and section-level [33][18]. polymorphic samples frequently encrypt code or produce dead structures to fool feature extraction procedures, and static features deteriorate quickly under severe obfuscation [12].

dynamic analysis tends to be more resistant to polymorphic transformations when combined with forensic artifacts like runtime behavior, API-call patterns, registry modifications, network traffic, and system interactions [3][20]. Since behavioral attributes are very consistent across different

mutated variants, dynamic analysis has a prime role in forensic investigations due to the need for reliable identification of artifacts. For that purpose, virtualized environments such as Cuckoo Sandbox and INetSim are standard tools for acquiring actionable forensic evidence [4].

Thus, hybrid models combining both static and dynamic features have the best accuracy by merging structural and behavioral indicators into one detection system. [27][28] . these introduce computational overhead, scalability and deployment problems in forensic laboratories[4].

Recent years have seen a significant shift towards ML to detecting polymorphic malware. Feature engineered methods that make use of RF, SVM, and XGBoost to perform well [2] [19]. The deep learning models helps malware image classification through CNNs, API-sequence modeling using LSTMs, and graph neural networks for control-flow and relational analysis, have mostly reported state-of-the-art results in the literature [13][8][23]. Despite their accuracy, deep models are limited by poor interpretability, challenging forensic transparency and court-admissible reporting [24][25].

Recent public datasets, such as, SOREL-20M, MaleX, and others, have insufficient labeling of polymorphic variants or lack obfuscation metadata, as discussed in [14][9]. This leads to poor reproducibility and the failure of machine-learning models to generalize when faced with real-world polymorphic malware, as discussed [26] [32]. This is why forensic readiness demands standardized datasets, transparent evaluation pipelines, and inherently interpretable machine learning.

A thorough review is necessary to map current detection techniques and identify forensic sound approaches, also to evaluate current strengths and weaknesses, and spot gaps that obstruct forensic investigations, given then advancement of malware and its significant impact on forensic workflows.

II. METHODOLOGY

The methodological framework given by Arksey and O'Malley in 2005 and the JBI Scoping Review Guidelines will serve as the base for the current review. The PRISMA-ScR checklist will serve as foundation for reporting structure that will ensure transparency, reproducibility, and clarity of the methodology.

Understanding the different detection techniques, analytical frameworks, and forensic techniques used in the identification /analysis of polymorphic malware variants was main objective of this review. The following three major research questions served as the foundation for this goal:

1) to determine the range of existing static, dynamic, hybrid, and machine-learning techniques;

2) examined the instruments, datasets, and forensic artifacts used in these various detection techniques;

3) established any existing methodological gaps or limitations from a forensic science perspective.

A structured search was performed in the academic databases such as IEEE, Elsevier, Springer, ACM Digital Library, Scopus, and Google Scholar. The search covered publications from January 2018 to January 2025 to make sure the review encompasses state-of-the-art developments in the detection of polymorphic malware. The development of search strings combined relevant words using Boolean operations: “polymorphic malware,” “obfuscated malware,” “dynamic malware analysis,” “static malware detection,” “hybrid malware detection,” “malware machine learning,” and “forensic malware analysis.” Other than database searching, backward citation tracking and reference list screening were also conducted to find those relevant studies that might not have been captured.

Studies were included if they were published in peer-reviewed journals or conference proceedings, focused on the polymorphic, metamorphic, obfuscated, or heavily packed malware, and presented a static, dynamic, hybrid or machine-learning-based detection approach. Eligible studies also needed to assess malware behaviour, forensic artefacts, or detection performance, include datasets or tools relevant to forensic malware analysis, and be published in English during the 2018–2025 period. The studies would be excluded if it focused exclusively on benign software obfuscation, did not have empirical evaluation, were not peer reviewed, examined general malware where relevance to polymorphism was not apparent, and the methodological detail was sparse such that it could not support forensic reproducibility.

PRISMA-ScR recommendations were used throughout the selection process. In total, 98 records were identified, and after the removal of 15 duplicates, a remaining total of 81 unique records remained. A total of 80 articles passed through to full-text assessment after title and abstract screening excluded one outdated or irrelevant study. After detailed assessment, a further nine studies were excluded for insufficient methodological detail, lack of empirical evidence, or limited forensic value. Thus, a total of 33 studies were deemed to meet all the eligibility criteria and therefore formed the final evidence base. A PRISMA-ScR flow diagram is recommended for presentation as Figure 1 to illustrate the selection process.

A structured charting form created specifically for this review was used to extract data. Information on the year of publication, malware classification, detection strategy, tools and datasets used, feature types, evaluation metrics used, and forensic relevance were gathered from each of the included studies. In order to highlight methodological trends, technological advancements, and current research gaps, the extracted data was then arranged into tables for comparison and thematic categories.

NFSU JOURNAL OF CYBER SECURITY & DIGITAL FORENSICS

NFSU – Journal of Cyber Security and Digital Forensics
Volume – 4, Issue – 2, December 2025
E – ISSN – 2583-7559

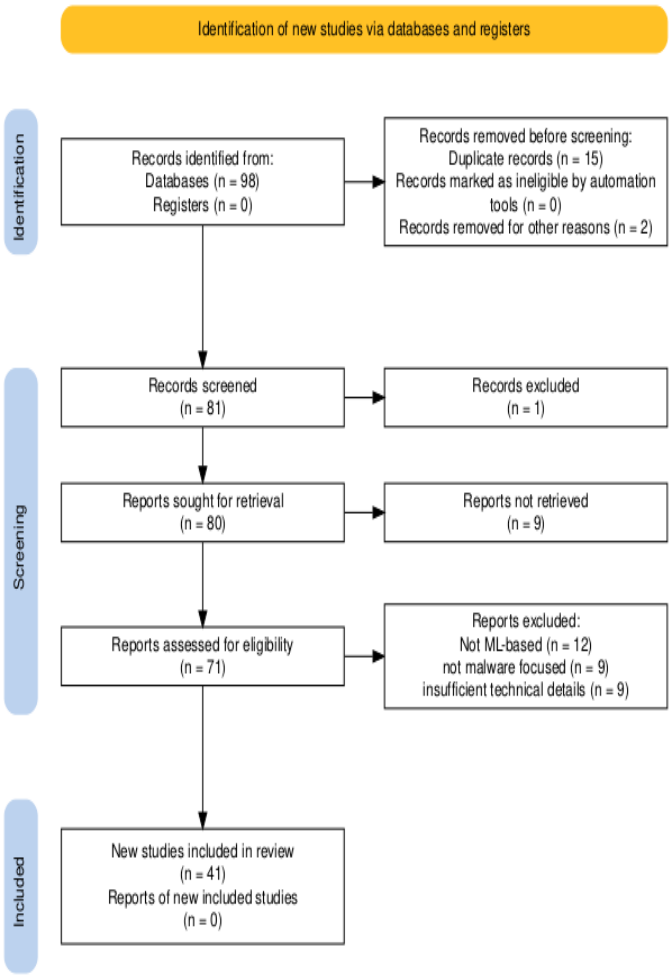


Figure 1 systemetic review flow diagram

The methodological diversity found in the included studies was managed using a narrative, thematic synthesis approach. Static analysis, dynamic behavior-based techniques, hybrid detection frameworks, and machine-learning or deep-learning approaches are the four main thematic headings under which we group the results. For each theme, the methods' advantages and disadvantages were examined in relation to how applicable they were to forensic applications. In order to investigate how elements like dataset selection, obfuscation complexity, and analysis environments may impact detection efficacy and forensic interpretability, comparisons between these studies were also carried out.

III. LITERATURE REVIEW

Polymorphic malware challenges in digital forensic investigations Because of its constantly changing code hides signs of compromise and makes timeline reconstruction more difficult and compromises the integrity of digital artifacts, and causes serious problems for digital forensic investigations. By offering a collection of analytical techniques that can understand behavioral, structural, and memory evidence even when malware variants differ in structure. The study's dynamic and hybrid approaches generate forensic relevant artifacts like network patterns, registry changes, API-call traces, and volatile memory changes , which will be utilized for threat actor identification , malwares activity reconstruction, and evidence integrity verification.

This review shows the roles that interpretable machine learning models and standardized evaluation ensuring that automated malware detection can be verified, replicated and accepted as forensic evidence in legal proceedings. It offers suggestions that will help forensic labs improve the dependability of digital evidence in investigations involving polymorphic threats by mapping gaps like dataset bias, anti-analysis behaviors, and lack of cross-environment validation. Because this work contributes knowledge that improves evidence acquisition, forensic interpretation, incident reconstruction, and legal defensibility during cybercrime investigations, it closely aligns with the goals of forensic science journals.

Malware creators are increasingly adopting polymorphic and metamorphic techniques to evade traditional antivirus systems. Unlike static signatures, polymorphic malware mutates its code by encrypting sections with a changing key or inserting junk instructions each time it propagates, while maintaining its malicious behavior [10]. This code polymorphism, and its more extreme form, makes detection very challenging, as this identical signatures no longer match new variants [3]. As a result, modern detection systems must include more sophisticated high end classifiers [2][33]. Polymorphic malware creators also target mobile and IoT platformst [32].

In recent years, researchers have explored a variety of methods to detect polymorphic malware using static code analysis, dynamic behavior analysis, or hybrid combinations [27] but all these methods are very time consuming as well as resource consuming so for faster and robust detection many researcher approaches machine learning and deep learning methods to capture subtle pattern that persist across variants [26][23]. recent technique involve image based binary analysis [12], API sequence modeling [1], GNN [13][23] and generative models for augmenting detection [30].

A. Static Analysis Techniques

Static analysis inspects the malware binary without executing it. Traditional static methods include signature scanning, but polymorphism undermines simple signatures. Therefore the extraction of robust static features from binary became crucial. Features such as opcode frequency histograms, byte n-gram distributions, and entropy measures were used [33] [2]. Another work converted binary into grayscale image and trained CNN (Convolutional Neural Networks) to classify[12]. Binary metadata characteristics (PE header parameters and section details) are employed as static inputs for classifiers

[33] Structural evaluations generate CFGs (control-flow graphs) from disassembled code and derive graph metrics; these graphs can be utilized directly within graph models or distilled into feature vectors [13][23]. After extraction, static attributes are introduced to classifiers, like RF(Random Forests), SVM (Support Vector Machines), or advanced deep learning ones; using ensemble classifiers is quite common and can boost overall resilience oftentimes [2][25]. The attraction to static analysis is its speed and its lack of dependency upon execution and thus its suitability to the analysis of large-scale corpora [14]. However, its disadvantages are clear: heavy obfuscation or code encryption can obscure code structure, and static analysis is ineffective at discovering memory unpacking or runtimes' behaviors, being oftentimes employed by polymorphic loaders [10][27].

B. Dynamic Behavioral Analysis:

Dynamic analysis analyzes malware behavior when it runs inside any controlled environment through recording API calls, system activities, changes to files and registries, and network communications. The footprints left during runtimes are oftentimes harder for polymorphic malware to completely hide, as it normally runs a variant with equal malicious conduct through analogous behavioral routines [3] [1].

Several studies openly model behavioral dynamism. HMM (Hidden Markov Models) have been applied to sequences of system calls or API calls to discover common sequences of malicious behavior; HMM-based detectors have shown excellent detection rates in experimental environments by using consistent behavior, even with code mutations[29]. RNN Like LSTM(Long Short-Term Memory) , that been have trained with execution traces like basic-block or API sequences and perform excellently on abstractions of these sequences [1]. The research proves that trace abstraction to basic blocks or higher-level actions often produces generalized performance superior to raw instruction-level traces [1]. Some of these also exploit network-traffic behavior by aggregating and coding malware network flows[15].

For network-centric areas (e.g., IoT), traffic feature and flow statistic-based dynamic methods are applied; federated learning and distributed LSTM ensembles have been raised to synchronize detection among edge gateways with privacy maintenance [23][30]. Dynamic platforms commonly utilized from literature are Cuckoo Sandbox and web services such as ANY.RUN for trace acquisition and automatic execution [14].

Dynamic analysis can identify activities hovering beneath static methods' radar but is itself plagued by disadvantages: it is very time-and resource-consuming, can necessitate a large number of execution traces to ensure coverage of behavior, and sophisticated malware can identify sandbox environments and quash malign activities [10]. These operational drawbacks justify integrating dynamic observations with other techniques whenever possible [27][25].

C. Hybrid Detection Approaches

Hybrid detection integrates static and dynamic characteristics to leverage their respective strengths. Some of these explored systems interlink static embeddings (CNN features extracted from code images at the bytecode level or code embeddings) with dynamic embeddings (RNN/LSTM encoding of API traces) to form aggregate vectors to classify and enhance immunity to polymorphic mutations [12][1][27]. Others intermingle structural static information (such as CFG metrics or graph embeddings) with run information (such as API counts and network flows) and submit the combined representation to ensemble learners or GNNs [13][23].

Multistage pipelines are very common in practice, whereby an efficient static filter identifies potentially malicious files, which then undergo execution in a sandbox environment for extended dynamic analysis. This allows for a reduction of the overall resources spent while retaining effectiveness in the detection. For instance, P and Rudra (2021) discuss that ensembles, which importantly combine static and dynamic classifiers through voting mechanisms or rule-based logics, normally exhibit higher detection rates along with reduced occurrences of false negatives. The complexity, laborious feature alignment, latency, and high operational demands upon scaling of hybrid systems raise concerns about their practicality [14][25].

D. Models of Machine Learning and Deep Learning

ML has become the dominant approach for polymorphic malware detection because of its capability of learning the invariant patterns across mutated variants. Classic classifiers such as Random Forest, SVM, KNN, and Logistic Regression are doing well on statically or dynamically engineered features. Ensemble models (Random Forest, XGBoost) performed consistently better than individual classifiers, notably when working with noisy or imbalanced datasets.[2] [19]

Deep learning methods allow for more automated feature extraction LIKE CNNs on binary images capture spatial structural patterns; temporal relations in execution sequences are learned by LSTMs and GRUs. The most recent developments underpin GNNs, which directly model the binary structure by embedding CFG or function graphs, and thus provide excellent robustness to polymorphic mutations. [13][23].Generative models-both GANs and Variational Autoencoders-increasingly find use in dataset augmentation and testing for adversarial robustness[30]. The models simulate new polymorphic variants, testing system stability in evolving threats.

E. Tools and Frameworks Used Across Studies

The literature makes extensive use of a combination of static and dynamic analysis tools and their use is summarized in table 1. Disassemblers such as Ghidra and Radare2 support

code inspection, instruction extraction, and CFG construction. Dynamic analysis environments—including Cuckoo Sandbox, ANY.RUN, VMware, and VirtualBox—enable secure execution and logging of system-level events. ML frameworks such as TensorFlow, PyTorch, and Scikit-learn underpin almost all learning-based detection systems. Code-embedding tools like Asm2Vec and SAFE generate neural representations of functions for graph-based models. Additional tools for sequence alignment (MAFFT, Smith–Waterman) and topological analysis (Ripser++) are used for specialised structural comparison tasks. The choice and configuration of these tools significantly influence experimental fidelity and reproducibility.

F. Datasets Used in studies

Studies rely on multiple sources of malicious and benign binaries. Public repositories such as VirusShare, VirusTotal, and MalwareBazaar provide large volumes of labelled malware samples, though polymorphic annotations are often incomplete[14][9]. Benchmark feature datasets such as EMBER and MalImg are widely used for static-method evaluations, while Android-focused studies employ Drebin or AndroZoo. In order to identify behavioral patterns, IoT-specific research analyzes network traces or device-level telemetry. To reduce false positives, many studies create custom benign corpora using open-source software. However, dataset bias is a persistent problem: synthetic corpora or unbalanced collections frequently inflate performance results and decrease generalization across real-world threats

Dataset Name / Source	**Malware Count**	**Benign Count**	**Features Provided**
Georgia Tech Apiary (COMSNETS 2022)	8,172 (6,595 active)	N/A	Dynamic replay logs, system calls, timestamps
VirusShare + GAN-augmented (Future Internet 2023)	8,152	1,000	Raw binaries, metadata (hashes, tags)
Kaggle PE dataset (Sci Reports 2025)	6,526	13,474	PE header, imports/exports, entropy, opcode n-grams
MALNET (SDS 2023)	1,262,024	N/A	Function-call graphs, binary images, hierarchical labels
Custom API trace corpus (AIP Conf. 2022)	1,220	N/A	API trace sequences, behavioral features
Windows API Trace Dataset (IEEE Access 2021, Acarturk et al.)	~9,000 (families)	N/A	Execution traces: instruction/basic-block sequences
API Call Sequence Dataset [3]	6,238	5,921	Ordered API names, timestamps

Hybrid Dataset (SN Computer Science 2021, P. & Rudra)	4,000	4,000	Mixed static and dynamic PE features
Selamat & Ali (Journal of Positive School Psychology 2022)	3,200	800	PE static with dynamic
HM3ALD (Applied Sciences 2018, Tajoddin & Jalili)	3,500	500	Program behavior traces, API sequences
Kavach App Dataset (IJARESM 2025)	N/A	N/A	Mobile malware app dataset
Chen et al. (TIFS 2023, GNN)	~45,000	~12,000	Feature vectors: PE header, imports, entropy, strings
Yousuf et al. (PeerJ CS 2023)	11,200	3,800	PE static features (headers, imports)
MalImg	~9,000 (families)	N/A	Grayscale images of binaries
EMBER (Elastic 2018)	600,000 (half benign)	600,000	JSON/CSV feature vectors
Drebin (Android)	5,600	5,000	Android APK permissions, intents, API calls
Microsoft Malware Classification Challenge (BIG 2015)	10,868	N/A	Bytes dump, ASM disassembly
MalwareBazaar (Abuse.ch)	>1M	N/A	Raw samples + metadata (imphash, family tags)

G. RESEARCH GAP

The lack of standard testing procedures is a research limitation. most of the studies evaluate models using train test splits, which unconsciously allow family level leakage between sets and effect accuracy. small number of people use informed validation, which is more like real world deployment as they used older samples to train the model and newer samples to test it. Comparing different papers is difficult as the diversity of datasets and experimental settings, also evaluation metrics listed in Table 3. Reproducibility is challenging cause dynamic analysis does not report their

execution environments or sandbox configurations. even though accuracy exceed 95%, these metrics need to be carefully understood because they may not accurately reflect live forensic workloads, large deployment or any adversarial conditions.

IV. *DISCUSSION*

This review surveyed 41 recent papers on polymorphic malware detection and found that analytical techniques have advanced clearly but still large gaps that limit their forensic application, in current analytical techniques, dynamic, hybrid, and machine-learning-based methods, have increased capability of detecting polymorphic malware, even though it has become more advance and robust [3][14][25].

A. Interpretation of Findings

The findings conforms that hybrid and dynamic analysis is the most effective to detect polymorphic obfuscation. Polymorphic malware often keeps behavioral invariants like API calls , registry manipulation, C2 interactions even when there is code mutation[3][1][15]. Dynamic detectors learning models like LSTM, can continue to overtake static ones due to these runtime signals. dynamic analysis generates rich forensic artifacts that are necessary for incident attribution and timeline reconstruction such as execution traces, volatile memory structures[14][30]

Despite being quick and scalable, encryption, multi-layer packing, and metamorphic transformations continue to undermine static analysis[10][12]. the situation is improved use of either control flow graphs, opcode, or image based representations but they still suffer from obfuscation[12][13]. Static analysis is therefore insufficient to support forensic investigations that require thorough evidentiary background[10][27].

Because a combination of structural and behavioral features leads to higher accuracy, fewer false negatives, and increased generalization across polymorphic variants, hybrid approaches perform the best overall[25][27]. Hybrid systems are positioned as the most operationally relevant frameworks for forensic practitioners due to their alignment with multi-artefact forensic workflows, where evidence must be correlated across logs, registry entries, memory dumps, and network traces.[14][27].

B. Methodological Weaknesses throughout the Literature

The majority of the reviewed studies still had a number of methodological flaws, despite encouraging developments. Complex polymorphic behavior has not been represented in the datasets utilized thus far[14][27]. Cross-variant robustness testing is limited by the lack of labeled variant lineages and well-defined mutation descriptors in publicly available datasets such as EMBER, SOREL-20M, and CIC-MalMem[14][9]. The majority of these studies present inflated detection metrics that do not accurately reflect adversarial real-world conditions because they primarily rely on synthetic or laboratory-generated datasets.[14][25]

Unreliable evaluations were frequent and Cross study comparisons are difficult also compromise reproducibility due to variations in splits, unreported sandbox settings and uncontrolled environmental conditions. In addition , only one of the research considered on evasion and anti forensic behaviors, which are found in real world forensic scenarios where malware can identify virtualization and postpone execution, or alter behavior[10].

The lack of transparency that many ML models have is also a flaw. In forensic and legal environments where reasoning, traceability, and clarity are crucial, black box designs lacking interpretable system are inappropriate[25]. despite their relevance to investigative, operation level attribution and family tracing have received little attention .When combined, these flaws a gap between the current realities of digital forensics and academic malware research.

C. FORENSIC SCIENCE IMPLICATIONS

Forensic scientists want to find a method of detecting malware that provides both high levels of accuracy and reliability of the evidence collected, many detection methods have been developed which provide higher accuracy in detecting malware[2][13]. But they produce results that are difficult to reproduce and are not easily taken as forensic evidence[24]. Because of this, many of the current detection methods lack the quality of evidence that required to be admitted into court[25].

Another problem exists with the ability to reproduce the findings of previous studies. Many studies that develop machine learning based malware detection systems evaluate using very controlled test conditions[14]. ML based systems are developed and tested using data sets that do not accurately represent all possible types of malware found in the wild. If the data used to train and test the models is not representative of the real world, then it will be impossible to independently verify the results of those studies and thus impossible to use them as credible evidence in court [25].

D. FUTURE DIRECTIONS

To help solve this problem there needs to be a development of detection systems that can be explained by the user, are strong to shifty actions taken by malware and are able to be validated across multiple different test environments and use data that has been made available in a transparent way, and can integrate seamlessly with existing digital forensic artifact extraction tools .

1) There are several obvious areas that need to be studied further. there is an urgent need for the development of benchmark datasets focused on polymorphic malware that

include information about the variant lineage, mutation metadata, behavioral labels, and obfuscation descriptors.

2) there is a need to incorporate XAI (explainable AI) into machine-learning-based detectors so that the interpretations produced by these models are suitable for forensic and legal purposes.

3) there is a need to validate these detection systems in a variety of sandbox configurations, operating system builds, and hardware profiles to determine how well they perform when operating in different environments.

4) the integration of ML/DL models with digital forensic tools and processes such as memory analysis, behavioral trace reconstruction, and automated timeline generation would greatly increase the efficiency of forensic examinations.

5) there is a greater need to evaluate these systems using real-world attacks that utilize evasion, anti-sandboxing, and obfuscation techniques.

Finally, new techniques for attributing malware families and reconstructing lineage graphs have shown great potential.

V. Conclusion

The review, from a forensic science perspective, mapped the current methods, analytical frameworks, and research trends of polymorphic malware detection. The summary of 33 peer-reviewed studies basically showed that while significant progress has been made in modeling the behavioral and structural patterns of polymorphic malware, there are still major obstacles in the way of creating reliable, repeatable, and forensically admissible detection systems.

Dynamic and hybrid approaches worked much better than static methods because they were able to capture behavioral invariants across different mutated variants. State-of-the-art machine learning and deep learning methods, represented by CNN-, LSTM-, transformer-, and graph-based models in automatic feature extraction, have advanced the accuracy of detection. However, much of this improvement still is due to the low interpretability of complex ML , insufficient evaluation procedures, and dataset limitations. This fact has important ramifications in terms of limits to legal defensibility, evidentiary transparency, and forensic reliability. For forensic practitioners, accurate and repeatable detection techniques, transparent in their operation, and resistant to adversarial manipulation are paramount. Actual deployments of these systems in forensic settings are severely hindered due to the lack of publicly available standardized polymorphic datasets, a general lack of cross-environment validation, and a lack of studies on evasion-resistant behaviors. If malware detection research is to be operationally relevant and scientifically sound, it needs to fix these flaws. Generating lineage-labeled, polymorphic datasets, embedding explainable AI into malware detection workflows, establishing evasion-aware testing environments, and utilizing standardized evaluation methodologies are the goals of future work. These advances have the potential to harden automated detection reliability in forensic contexts and increase its accuracy.

Thus, this review emphasizes the need to bring academic malware research in line with practical needs in the domain of digital forensics. In this way, next-generation detection models could provide the technical precision and forensic resilience required for critical investigative and legal procedures.

References

[1] Acarturk, M. Sirlanci, P. G. Balikcioglu, D. Demirci, N. Sahin, and O. A. Kucuk, “Malicious code detection: Run trace output analysis by LSTM,” *IEEE Access*, vol. 9, pp. 9625–9635, 2021, doi: 10.1109/ACCESS.2021.3049200.

[2] M. S. Akhtar and T. Feng, “Malware analysis and detection using machine learning algorithms,” *Symmetry*, vol. 14, no. 11, p. 2304, 2022, doi: 10.3390/sym14112304.

[3] E. Amer and I. Zelinka, “A dynamic Windows malware detection and prediction method based on contextual understanding of API call sequence,” *Computers & Security*, vol. 92, p. 101760, 2020, doi: 10.1016/j.cose.2020.101760.

[4] J. Ananth, M. Kumaran, and J. L. E. Chandra, “Behavioral analysis and machine learning for polymorphic malware detection and classification,” *Int. J. Adv. Res. Comput. Commun. Eng.*, 2023, doi: 10.17148/IJARCCE.2023.12687.

[5] H. Arksey and L. O’Malley, “Scoping studies: Towards a methodological framework,” *Int. J. Social Research Methodology*, vol. 8, no. 1, pp. 19–32, 2005, doi: 10.1080/1364557032000119616.

[6] A. Augello, A. De Paola, and G. Lo Re, “Hybrid multilevel detection of mobile devices malware under concept drift,” *J. Netw. Syst. Manage.*, vol. 33, p. 36, 2025, doi: 10.1007/s10922-025-09906-3.

[7] M. Avhankar, J. Pawar, and V. Kumbhar, “A comprehensive survey on polymorphic malware analysis: Challenges, techniques, and future directions,” *Communications on Applied Nonlinear Analysis*, 2025, doi: 10.52783/cana.v32.4554.

[8] A. Bensaoud and J. Kalita, “CNN-LSTM and transfer learning models for malware classification based on opcodes and API calls,” *Knowledge-Based Systems*, vol. 290, 2024, doi: 10.48550/arXiv.2405.02548.

[9] Camargo, R. C., Nieto, J. M., Rojas, N., Díaz-López, D., Alférez, S., Gómez, A. L. P., Nespoli, P., Mármol, F. G., & Karabiyik, U. (2025). DEFENDIFY: defense amplified with transfer learning for obfuscated malware

framework. *Cybersecurity*, *8*(1). https://doi.org/10.1186/s42400-025-00396-z.

[10] C. Catalano, A. Chezzi, M. Angelelli, and F. Tommasi, "Deceiving AI-based malware detection through polymorphic attacks," *Computers in Industry*, vol. 143, p. 103751, 2022, doi: 10.1016/j.compind.2022.103751.

[11] M. Chaikovskyi *et al.*, "Comprehensive approach to the detection and analysis of polymorphic malware," in *Proc. ICyberPhyS 2024*, CEUR Workshop Proc., pp. 1–11, 2023.

[12] A. Chakraborty, K. Kriti, N. Yateendra, and B. Praba, "Polymorphic malware detection by image conversion technique," *Int. J. Eng. Adv. Technol.*, vol. 9, no. 3, pp. 2898–2903, 2020, doi: 10.35940/ijeat.b4999.029320.

[13] Chen, Y., Lin, S., Huang, S., Lei, C., & Huang, C. (2023). Guided malware sample analysis based on graph neural networks. *IEEE Transactions on Information Forensics and Security*, *18*, 4128–4143. https://doi.org/10.1109/tifs.2023.3283913.

[14] S. Dambra *et al.*, "Decoding the secrets of machine learning in malware classification," in *Proc. ACM CCS*, pp. 60–74, 2023, doi: 10.1145/3576915.3616589.

[15] X. Deng and J. Mirkovic, "Polymorphic malware behavior through network trace analysis," in *Proc. COMSNETS*, pp. 138–146, 2022, doi: 10.1109/COMSNETS53615.2022.9668396.

[16] J. Drew, M. Hahsler, and T. Moore, "Polymorphic malware detection using sequence classification methods and ensembles," *EURASIP J. Inf. Security*, p. 2, 2017, doi: 10.1186/s13635-017-0055-6.

[17] Gazzan, M., Alobaywi, B., Almutairi, M., & Sheldon, F. T. (2025). A deep learning framework for enhanced detection of polymorphic ransomware. *Future Internet*, *17*(7), 311. https://doi.org/10.3390/fi17070311.

[18] F. Hamid, "Enhancing malware detection with static analysis using machine learning," *Int. J. Res. Appl. Sci. Eng. Technol.*, vol. 7, pp. 38–42, 2019, doi: 10.22214/ijraset.2019.6010.

[19] R. Kandakatla and K. Rajakumari, "Machine learning techniques for polymorphic malware analysis and identification," *Int. J. Innov. Sci. Res. Technol.*, vol. 9, no. 11, pp. 1496–1498, 2024, doi: 10.38124/IJISRT24NOV1395.

[20] A. Karakaya and A. Ulu, "Dynamic malware detection approach based on API calls," *Int. J. Inf. Security Sci.*, 2024, doi: 10.55859/ijiss.1510423.

[21] Krasser, S., Spurlock, J., Radu, M., Moon, B., Korn, A., Seth, M., & Bausewein, C. (2024). Machine Learning-Based malware detection in a production setting. In *Advances in information security* (pp. 119–142). https://doi.org/10.1007/978-3-031-66245-4_5.

[22] Kuriyal, V., Bordoloi, D., Singh, D. P., & Tripathi, V. (2022). Metamorphic and polymorphic malware detection and classification using dynamic analysis of API calls. *AIP Conference Proceedings*, *2767*, 020029. https://doi.org/10.1063/5.0104235.

[23] Li, T., Luo, Y., Wan, X., Li, Q., Liu, Q., Wang, R., Jia, C., & Xiao, Y. (2024). A malware detection model based on imbalanced heterogeneous graph embeddings. *Expert Systems With Applications*, *246*, 123109. https://doi.org/10.1016/j.eswa.2023.123109.

[24] Madamidola, O. A., Ngobigha, F., & Ez-Zizi, A. (2024). Detecting new obfuscated malware variants: A lightweight and interpretable machine learning approach. *Intelligent Systems With Applications*, *25*, 200472. https://doi.org/10.1016/j.iswa.2024.200472.

[25] L. Mauri and E. Damiani, "Hardening behavioral classifiers against polymorphic malware," *Information Sciences*, vol. 689, p. 121499, 2024, doi: 10.1016/j.ins.2024.121499.

[26] M. Miraoui and M. B. Belgacem, "Binary and multiclass malware classification of Windows PE files," *Frontiers in Computer Science*, vol. 7, 2025, doi: 10.3389/fcomp.2025.1539519.

[27] A. P. and B. Rudra, "Study of a hybrid approach towards malware detection in executable files," *SN Comput. Sci.*, vol. 2, no. 4, 2021, doi: 10.1007/s42979-021-00672-y.

[28] N. S. Selamat, "Al FHM polymorphic malware detection based on supervised machine learning," *J. Positive School Psychology*, vol. 6, no. 3, pp. 8538–8547, 2022.

[29] A. Tajoddin and S. Jalili, "HM3ALD: Polymorphic malware detection using program behavior-aware hidden Markov model," *Applied Sciences*, vol. 8, no. 7, p. 1044, 2018, doi: 10.3390/app8071044.

[30] Urooj, U., Al-Rimy, B. a. S., Zainal, A. B., Saeed, F., Abdelmaboud, A., & Nagmeldin, W. (2023). Addressing Behavioral Drift in Ransomware Early Detection Through Weighted Generative Adversarial Networks. *IEEE Access*, *12*, 3910–3925. https://doi.org/10.1109/access.2023.3348451.

[31] N. K. Verma, "Multiclass malware detection in operational technology systems," *J. Inf. Syst. Eng. Manage.*, vol. 10, no. 3, pp. 292–308, 2025, doi: 10.52783/jisem.v10i3.4546.

[32] S. Y. Yerima, "High accuracy detection of mobile malware using machine learning," *Electronics*, vol. 12, no. 6, p. 1408, 2023, doi: 10.3390/electronics12061408.

[33] Yousuf MI, Anwer I, Riasat A, Zia KT, Kim S. 2023. Windows malware detection based on static analysis with multiple features. *PeerJ Computer Science* 9:e1319 https://doi.org/10.7717/peerj-cs.1319

[34] Fiadufe, F., Modi, K., Shukla, K. and Etyang, F.O. (2025) 'Forensic investigation and analysis of malware in Windows OS', *Int. J. Electronic Security and Digital Forensics*, Vol. 17, Nos. 1/2, pp.169–182., doi: 10.1504/IJESDF.2025.143477.